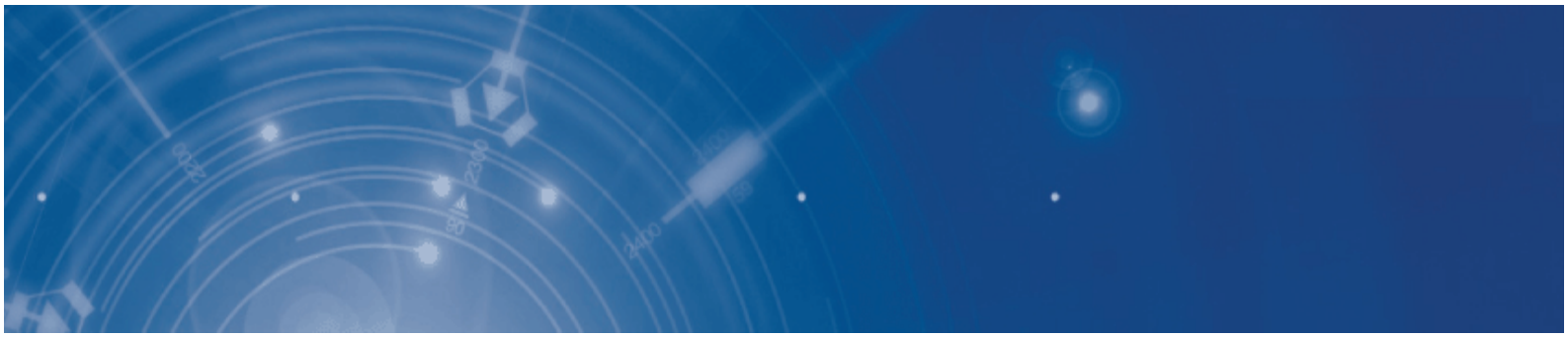


Gartner

The Emerging Architecture of Identity Management

Assessment (Single Instance Use Case)

Bob Blakley

Identity and Privacy Strategies

Identity management (IdM) is not aging gracefully. Enterprise IdM systems, designed to centralize management of the information used to authenticate employees and authorize their access to enterprise resources, have matured. Unfortunately, the business environment in which centralized management of identity made sense is fading away just as the technology necessary to support it has become widely available. Tweaks to the existing IdM architecture will not solve these problems. What's needed — and emerging — is a new IdM architecture based on new principles. In this assessment, Identity and Privacy Strategies Research Director Bob Blakley lays out the roadmap to this new architecture.

Summary of Findings

Bottom Line: Identity management (IdM) today is based on a push model — a model that was designed to retrofit centralized IdM into applications that were designed for stand-alone administration. The push model won't work in a world of many identity providers and widespread federation. There are compelling business reasons to move to a new IdM architecture, and that architecture is already emerging. It will be based not on pushing information from a central source at the time information changes, but instead, on pulling information to the point of use at the time of use.

Context: IdM is too expensive and not effective enough because the IdM problem has changed while the solution has not.

The old problem was to distribute information about enterprise employees from a central authoritative identity repository owned by the CIO to enterprise application systems also owned by the CIO, to support authentication and authorization of employees.

The new problem is to gather information about enterprise employees, contractors, partner personnel, and customers from multiple identity repositories, some owned by third parties, and distribute that information to enterprise and third-party applications, some hosted on-premises and some by third parties or in the cloud, to support use of enterprise applications by employees and third parties and use of third-party applications by enterprise personnel.

Takeaways:

- IdM isn't meeting the needs of federated enterprises.
- Federation has broken down the walls of enterprise and consumer identity silos.
- Cheap-but-weak consumer identities want to compete in a market against strong-but-expensive enterprise

identities.

- The current push-model IdM architecture can't support the development of a market for identities.
- Push-model IdM is a market for IdM systems.
- User-centric IdM is a market for identity providers.
- Businesses need a market for identities, and only a pull-model IdM architecture will allow such a market to emerge.
- The foundations for a pull-model IdM architecture already exist.
- But the pieces won't be put together into a coherent whole for several years.
- The transition will come in two phases.
    1. Production of identities will be separated from consumption of identities through the introduction of a virtual directory interface.
    2. Applications will externalize authorization to policy decision points (mostly Extensible Access Control Markup Language [XACML]-based), which can use contextual authorization to request attributes in real time.
- Enterprises should follow a six-phase roadmap to manage the transition to pull-model IdM:
    1. Divide production of identities from consumption
    2. Build an identity consumption policy
    3. Build support for multiple providers
    4. Externalize authorization from applications
    5. Contextualize authorization
    6. Phase out connectors

Return to Top

Analysis

Identity management (IdM) technology is not aging gracefully. Enterprise IdM systems, which were designed to centralize management of the information used to authenticate employees and authorize their access to enterprise resources, have finally matured. Unfortunately, the business environment in which centralized management of identity made sense has started to fade away just as the technology necessary to support it has become widely available.

Tweaks to the existing IdM architecture will not meet the emerging identity management requirements needs of the enterprise; today's IdM architecture was designed to solve an obsolete problem.

Return to Top

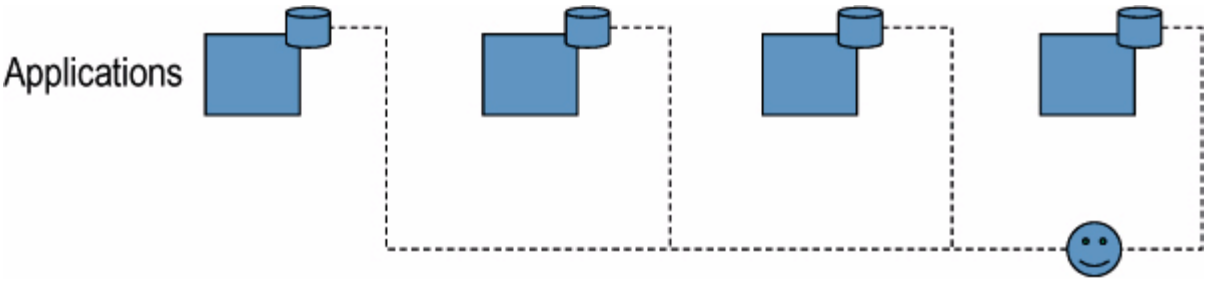
The Past

Yesterday's IdM problem:

> Distribute information about full-time enterprise employees from a central authoritative identity repository owned and operated by the CIO to a set of enterprise application systems also owned and operated by the CIO in order to support authentication and authorization of employees to applications.

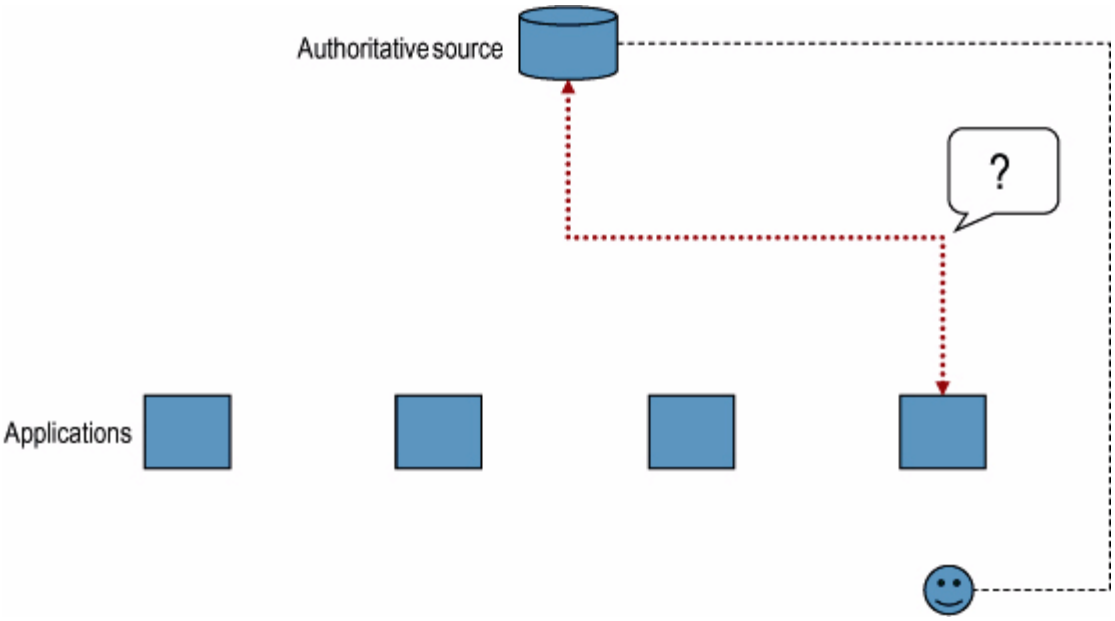
This problem was important because, as Figure 1 shows, enterprise IT grew up in silos; applications were originally designed to be locally administered, with no ability to receive user account information from a central source.

Figure 1. Enterprise IT Application Silos



As enterprise workers used more and more applications, administration of user accounts across the silos consumed more and more administrative effort and expense. As enterprise applications proliferated, enterprises began to automate HR administration. This led to the situation depicted in Figure 2: The enterprise had an authoritative source of information about employees, but it did not have a way to get this information from the authoritative source to the applications in which users needed accounts.

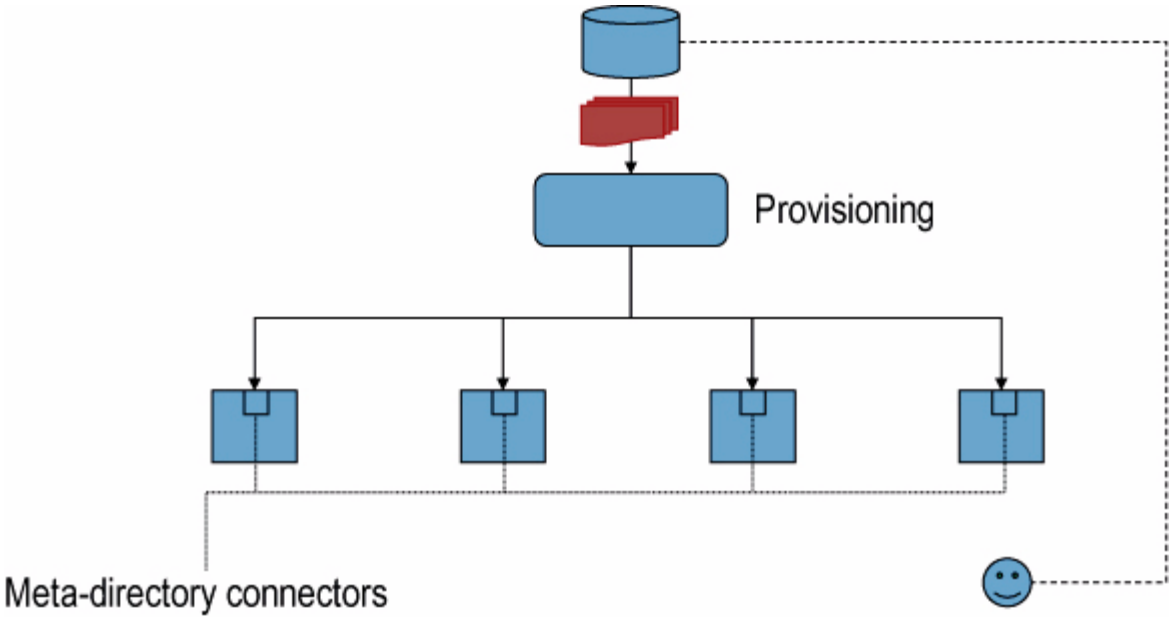
Figure 2. An Authoritative Identity Source Disconnected from Application Silos



As more and more employees began to use multiple applications, demand grew for integration. Users expressed the desire for integration as a requirement for single sign-on, and single sign-on solutions soon emerged. The simplest of these were essentially the electronic equivalent of sticky notes: client-side password databases coupled with scripts that fed the stored passwords to applications. These primitive solutions weren't great for users, and they were even worse for administrators; they provided no facilities for provisioning new cross-application account passwords to users, and they frequently made password change and password recovery processes more complex. Enterprises soon recognized a requirement for back-end account harmonization and provisioning.

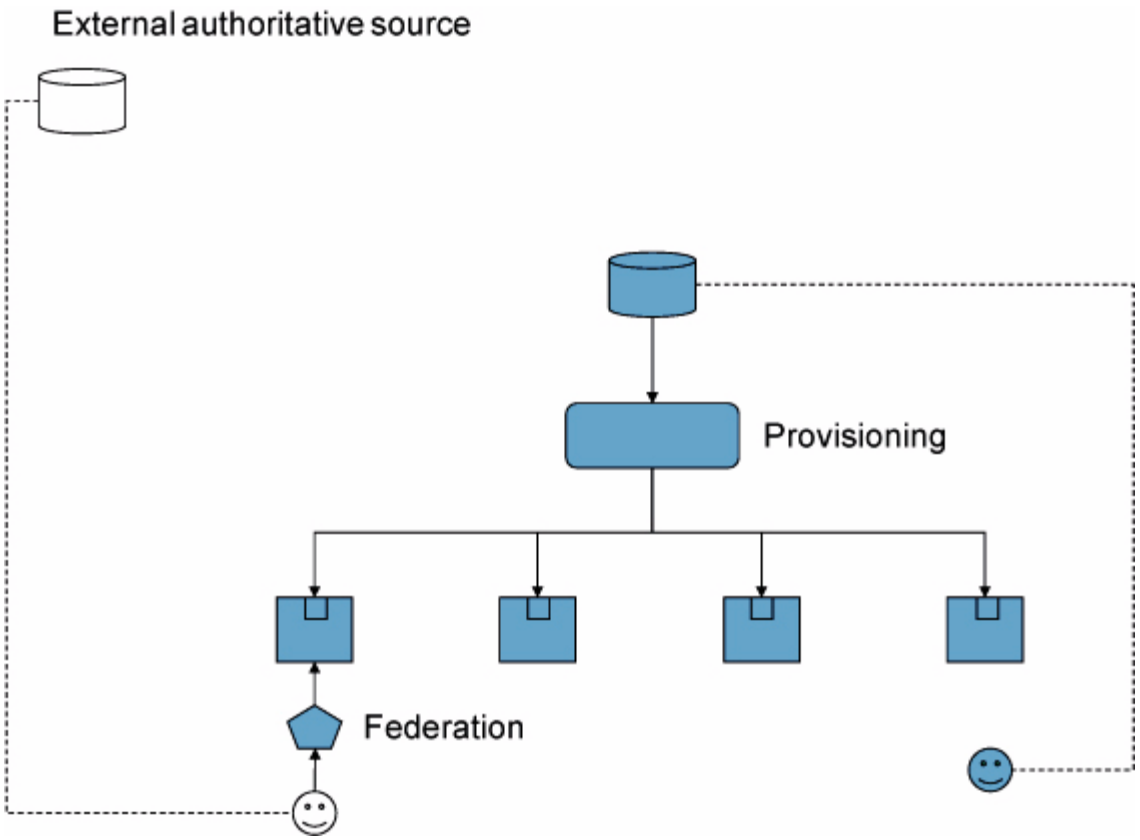
Early provisioning solutions were fine in simple, relatively homogeneous environments, but account provisioning and single sign-on in heterogeneous environments required solutions that were platform-neutral. Solving the heterogeneous account provisioning problem required a new invention: the provisioning connector. As Figure 3 illustrates, the provisioning connector (often implemented as a component of a meta-directory) served two purposes: it translated the data format of the authoritative repository into that required by the specific application to which identity information needed to be provisioned, and it implemented a protocol that connected the local administrative interface of the application to the data stream being pushed from the authoritative source to the application by the provisioning system.

Figure 3. Connector-Based Heterogeneous Provisioning



Provisioning connectors have proved to be a decidedly mixed blessing; they do the job they're designed for at the cost of complex integration (to authorize the connector to the application and to map the data schema of the authoritative source to that of the application) and of fragility (when an application or authoritative source is updated, it's often necessary to update the connector that mediates their provisioning relationship). But despite these issues, connector-based provisioning largely met the IdM needs of enterprises until federation entered the picture.

Figure 4. Federated Authentication Without Federated Provisioning



Federation technology was designed to meet a new business need to allow non-employees to access enterprises' internal IT systems. Three factors drove the need for federation: supply-chain automation and partnerships interconnected businesses' information systems, electronic commerce increasingly exposed business systems to customers, and outsourcing pressures forced businesses to rely on partner and contractor personnel for increasingly critical tasks.

Simple federation has enabled partner and contractor personnel to log on to enterprise systems, but it does not yet provide CISOs with the identity attribute information they need to control these people's access to enterprise systems; CISOs are left feeling as if they let a bunch of people into a bank whose safe-deposit boxes have no doors or locks.

The architecture illustrated in Figure 4 isn't working well in today's business environment. Some stresses on the IdM architecture have been clear for years; others have arisen more recently. These stresses include:

- **Cost:** IdM systems are expensive to buy, expensive to integrate, expensive to operate, and expensive to maintain. And they keep growing; each time an IdM project is completed, another requirement arises that requires acquisition, implementation, and operation of a new module or product.
- **Complexity:** IdM systems have a lot of moving parts, many of which (e.g., provisioning connectors) are fragile in the face of change in other parts of the system. Integration of IdM into an enterprise's IT infrastructure is almost always a complex, expensive, and multi-quarter project.
- Fine-grained access control: IdM systems are good at managing accounts and group memberships; they're not so good at managing roles, and their support for fine-grained entitlements is primitive.
- **Federation:** IdM systems are now pretty good at authenticating users across domain and organizational boundaries. They are not good at authorizing users across these boundaries; support for federated provisioning and identity attribute exchange across organizations is rudimentary.
- **Agility:** IdM systems work well in organizations with stable IT infrastructures, user populations, and management processes. Organizations that frequently change role descriptions, applications, and approval processes, or that experience frequent or large changes in user populations, create significant challenges for IdM systems: connector integration, data synchronization, workflow and schema management, and policy and compliance management.
- **Heterogeneity of population:** IdM systems work well in organizations that have a small number of in-house authoritative sources for identity information. Organizations that have multiple authoritative sources of identity information for individual users — or that have to support users whose authoritative identity information sources are third-party organizations with whom policies and technology choices must be negotiated — find it difficult to use IdM technology.
- **Decentralization:** In organizations with large populations of partners and contractors — or other users whose identities are managed by third parties — have a hard time using IdM technologies because many of the events which drive the identity lifecycle happen in other organizations and are therefore invisible to the on-premises IdM system and its operators.

The essential cause of the problem with today's architecture is "push." "Push" is the assumption — at the core of the IdM architecture — that identity information comes from one single central source, and before it can be used it has to be "pushed" to the point where it will be consumed. The assumption was logical when it was made; the organizations that needed to do IdM were highly centralized, so a single authoritative source (usually an HR system or enterprise directory) existed. And data had to be pushed to endpoints — the endpoints had been designed to be stand-alone systems, so they weren't designed to reach out to another system and ask for information they didn't have. Each endpoint assumed that its local administrator would use local interfaces to enter all the necessary identity information. The provisioning connector — a box that could receive data pushed from the authoritative repository and drop it into the administrative interface of an endpoint system — arose naturally in this environment and became the cornerstone of the modern IdM architecture.

Return to Top

The Present

The push-model IdM architecture is struggling because the IdM problem has changed. Compared with the problem the current IdM architecture was invented to solve, today's IdM problem is larger and more complicated:

> Gather information about enterprise employees, contractors, partner personnel, customers, and members of the general public from a variety of authoritative identity repositories, some on-premises and some owned and operated by third parties, and distribute that information to enterprise applications and third-party applications, some hosted on-premises and some hosted by third parties or in the cloud, in order to support use of enterprise applications by employees and third parties and use of third-party applications by enterprise employees and others operating on the enterprise's behalf.

The push-model IdM architecture can't solve this problem because it depends on the truth assumptions that are simply false in today's environment. Among these false assumptions are:

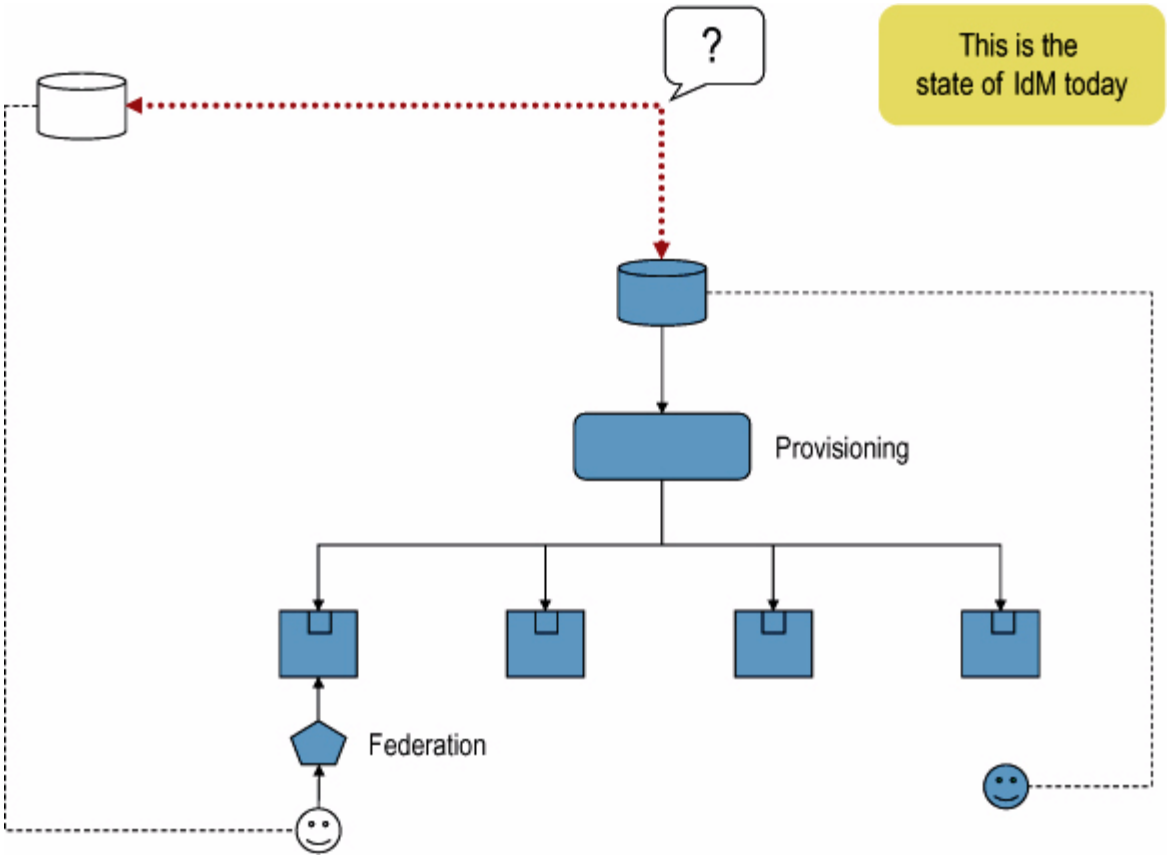
- Identity information comes from one place and goes to many places.
- Identity information will come from the same place tomorrow that it comes from today.
- Identity information comes from the same place no matter who the subject is.
- Identity information comes from the same place no matter what the attribute is.
- Identity information comes from within my enterprise — so the organization is responsible for identity assurance and can change it at will.
- Copies of identity information are sent to and stored by every system that might someday need to use the information.
- The places identity information is sent to and stored were not designed to be aware of IdM; they were designed to get identity information through local administration interfaces.
- If identity information needs to be retrieved from a new repository, the IT staff can make administrative and even possibly technical changes to that repository.
- If identity information needs to be provisioned into a new endpoint, the IT staff can make administrative and even possibly technical changes to that endpoint.
- The business is only dealing with half a million identities, so it can resynchronize all changes to all identities with all endpoints every hour/night/weekend.
- The business can easily and economically get accurate identity information about all the people who need to be authorized to use systems it cares about.

Trying to solve today's IdM problem with a push-model solution starting from these assumptions will produce a system in which all information about every user is transmitted from every organization to every other organization every night. Implementing user-centric technology will force every user to personally approve each and every one of these transfers. Needless to say, the result will be a usability disaster, a performance disaster, a data integrity disaster, a security disaster, a management disaster, a policy disaster, and many other kinds of disasters.

But that's where we're headed if we don't change course. And we're not changing course; IdM experts today argue either that we should fix the federated provisioning problem by adding another layer of "push" from off-premises repositories (as illustrated in Figure 5) or that we should abandon federated provisioning in favor of a user-centric model in which individuals push their own identity attributes to applications at resource access time.

Essentially this is an argument about whether third-party identity providers should push data through the enterprise's back door (e.g., using Service Provisioning Markup Language [SPML]-based federated provisioning) or through its front door (e.g., using information cards and Security Assertion Markup Language [SAML] tokens).

Figure 5. The Wrong Question: How Do We Push Across Domain Boundaries?



Return to Top

The Market

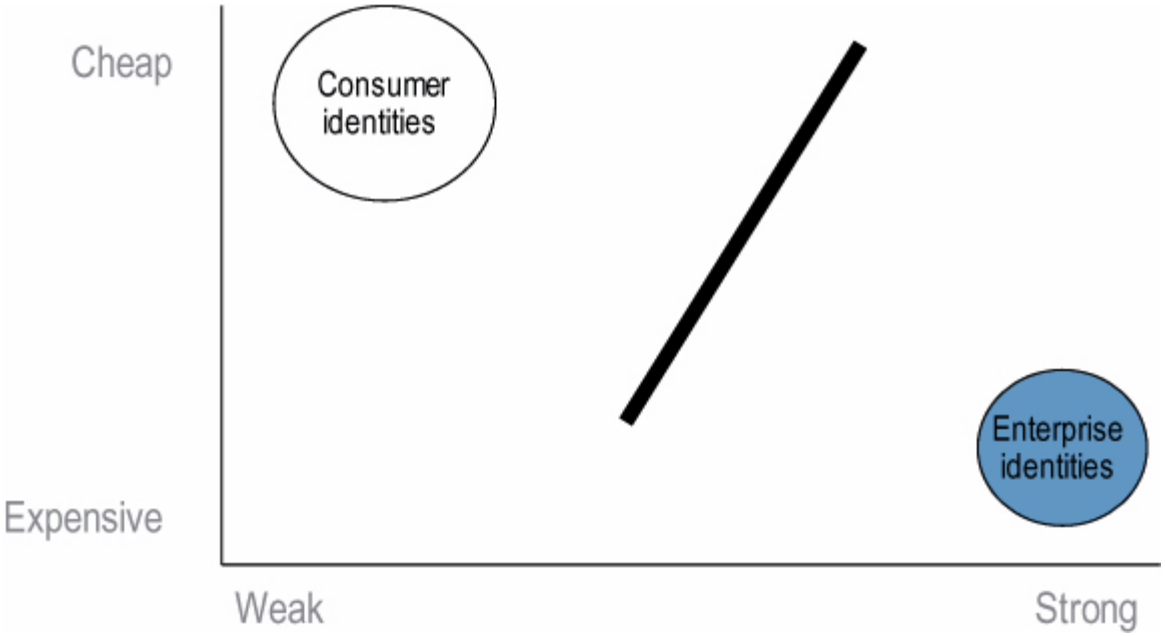
The real question federation raises is not "Should we push the data through the enterprise's back door or through its front door?"; the real question is "Now that we have federation, where should the enterprise get its identity information from, and how should it get the information?"

Until recently, enterprises had no choice in the matter. Most people had no digital identities, so if the enterprise wanted a person to use a digital identity, the enterprise had to create an identity, give it to the person, and manage it. This situation is illustrated in Figure 6; enterprises of the 1980s spent a lot of money on background checks, HR systems, helpdesk staffs, and enterprise information technology to build and maintain identities that their employees used to access on-premises applications.

Figure 6. An Enterprise Creating and Managing Its Own Identities

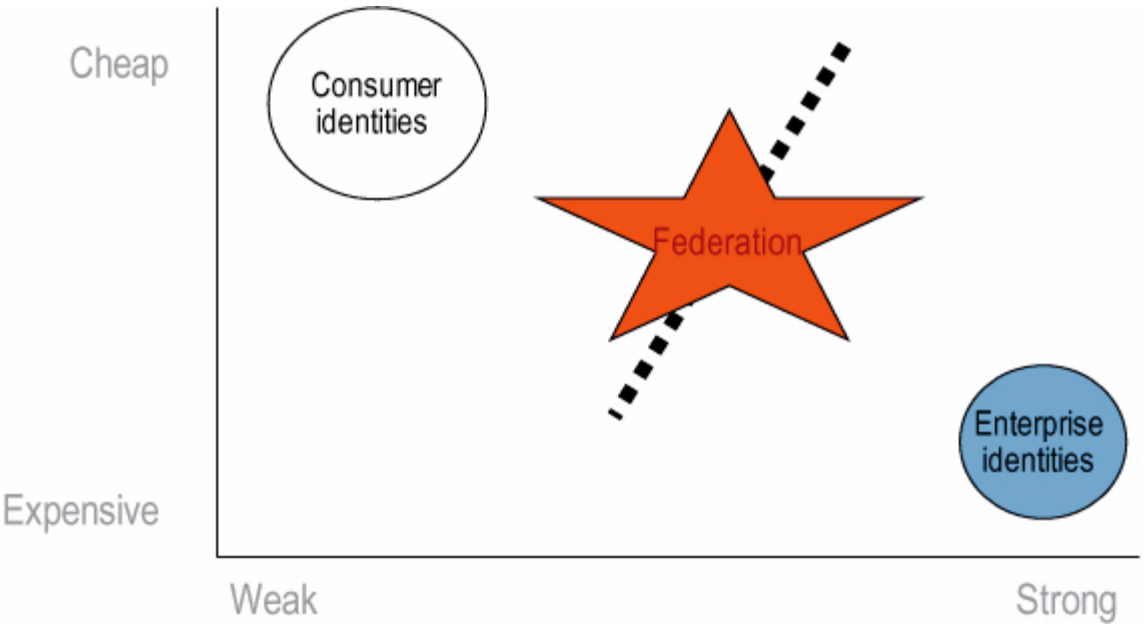
In 1980, most people had no electronic identities, and most of those who did have electronic identities got them at work and used them at work. That changed in the early 1980s when people began to pay to have accounts with consumer online services like CompuServe and AOL. These early consumer online services and their modern equivalents (e.g., Gmail, Facebook, Twitter, LinkedIn, MySpace, and others) did little, if any, background investigation and provided few security controls (compared with enterprise IdM systems) to protect consumer accounts against hijacking or other abuse. By 1990, as Figure 7 shows, many people had several cheap-but-weak consumer identities as well as one or more strong-but-expensive enterprise IDs.

Figure 7. Separate Identity Universes



The advent of consumer identities didn't affect enterprise IdM much at first because consumer IDs couldn't be used within the enterprise, and enterprise IDs couldn't be used to access consumer services; Figure 8 shows the wall between the world of consumer and enterprise identity. The loosening of restrictions on use of the Internet at work in the early 1990s meant that some enterprise employees used their consumer IDs from work systems, but for the most part, enterprise systems still used enterprise IDs and consumer systems still used consumer IDs-until federation came along.

Figure 8. Federation Shatters the Wall Between Consumer and Enterprise Identity



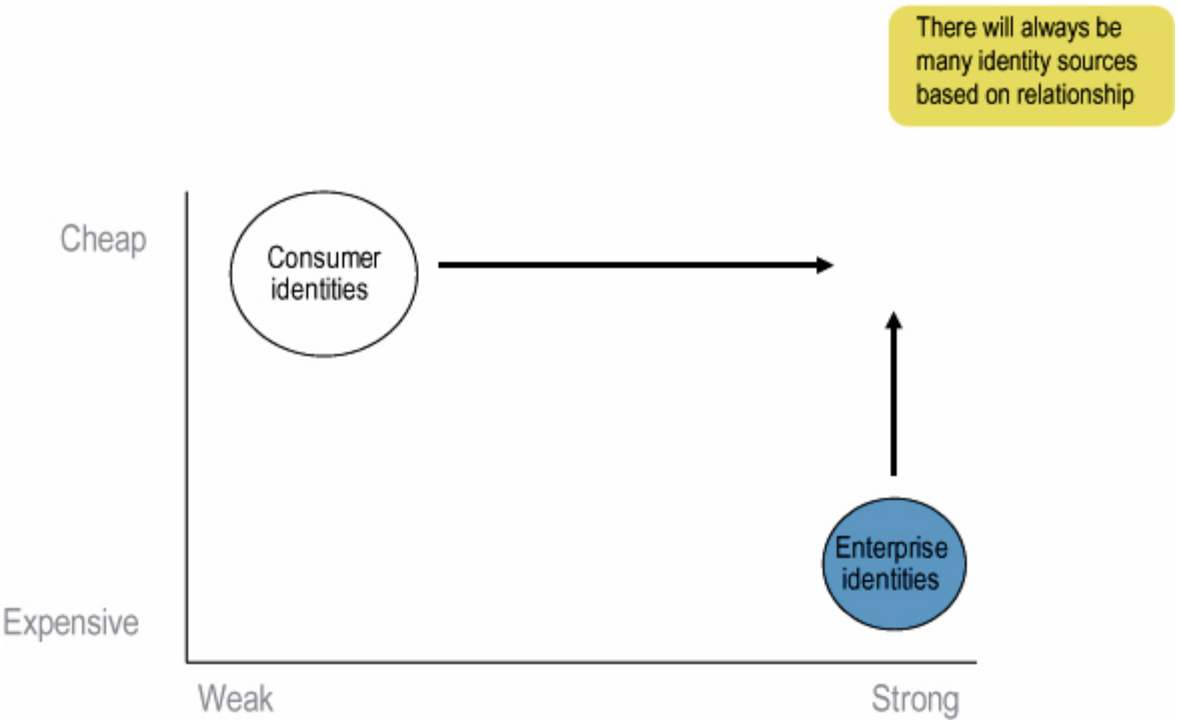
Although it wasn't immediately apparent, federation — and in particular the standardization of SAML and Web Services Trust Language (WS-Trust) as the bases for open federation standards during the period between 2002

and 2007 — broke down the wall between enterprise and consumer identities by allowing applications to consume identities produced anywhere.

The effect of eliminating the wall between enterprise and consumer identities is just starting to be felt, but it will be profound. Eliminating the wall will transform a landscape of identity silos into the environment shown in Figure 9. Indeed, the transformation is already under way. Google is forming partnerships to extend Google identities into the enterprise, and Salesforce.com is doing the same. Facebook aspires to be the central source of identity on the web, and no one should harbor the illusion that their ambition stops at the edge of the enterprise. And born-in-the-cloud identity providers such as Symplified are offering to replace on-premises IdM with their services.

Consumer identities are still generally weak, but they are cheap. Enterprise identities are strong, but they are expensive. Enterprises now have a choice: spend a lot of money in-house for strong identity or trade cost reduction for increased risk by going with an outside provider. The choice is putting pressure on all parties. CIOs' budgets for IdM are under pressure, and cloud and consumer identity providers are having their feet held to the identity assurance fire by customers who want higher-quality IDs.

Figure 9. A Market for Identities



Today's push-model IdM infrastructure doesn't allow enterprise identities and consumer identities to compete in an effective market, though this is what they would naturally do and it's what enterprises want them to do. The push-model identity environment stifles competition by locking enterprises into a small number of on-premises or third-party identity providers. The lock-in happens because of the overhead of switching providers; when an enterprise first selects an IdM system, it:

- Issues a request for proposal [RFP]
- Evaluates proposals
- Selects a provider
- Deploys and integrates a set of connectors to receive the provider's pushed data
- Enrolls all its users in the provider's system
- Pushes the data out to its endpoints
- Waits for incremental updates

Switching providers means doing it all over again: deploying and configuring new policy and new connectors, re-enrolling all the users in the new provider's system (possibly after removing them from the original provider's system), and pushing every user's identity data out to all endpoints. It's a massive effort that is accomplished at great expense.

User-centric identity isn't an economically viable alternative to this backdoor-bulk-push environment because, if users get exclusive control over what provider will be used for their information, the enterprise loses all control over provider quality and cost of identity provision.

Push-model IdM and user-centric identity both fall short of enterprise requirements for business reasons rather than technical reasons:

- Push-model IdM creates a market for *IdM systems*
- User-centric identity creates a market for *identity providers*

But enterprises don't need a market for IdM systems or a market for identity providers; they need a market for *identities*.

A market for *IdM systems* doesn't meet enterprise requirements for low-cost, high-quality identity because an IdM system isn't a factory that *produces* identities; it's a warehouse that *stores and distributes* identities. An IdM system only works if it's hooked up to an identity source — and it's the source (or sources) of identities, not the IdM system, that determines the enterprise's cost and quality of identity.

A market for *identity providers* doesn't meet enterprise requirements because no single source is authoritative for all the identity attributes a modern enterprise needs know about an individual (one source may be authoritative for credit information, another for citizenship and residency status, another for criminal record, another for professional licensure status, and so on) and because no source is authoritative for all the individuals an enterprise needs to identify. Choosing any single source or fixed collection of sources therefore guarantees that the enterprise will be buying some high-cost or low-quality identities.

A market for *identities* — which does not yet exist, but which is coming into existence — will allow enterprises to get identity information about each individual from the provider, whose balance of cost and quality for that specific individual is best at the specific moment the identity information is needed. Such a market can only exist if the enterprise can select an identity provider for a particular individual at the time of use based on a spot price and publicly stated assurance and business terms. Only a market for identities can provide quality identities at low cost because, as noted in the Burton Group overview "A Relationship Layer for the Web . . . and for Enterprises, Too," identities can only be provided at low cost and high quality by organizations that have close, good relationships with the subjects of those identities — and no single organization has close, good relationships with every person an enterprise needs to identify.

Building a market for identities will require enabling enterprises to get identity information from any provider at any moment — and a push model won't do that. A market for identities requires a pull-model infrastructure.

David Siegel's recent book *Pull: The Power of the Semantic Web to Transform Your Business* sets out some principles for a pull-model system:

- Data migrates from the deep (proprietary) web to open web
- Location doesn't matter, names do
- Data does not move (after it gets to the open web)

This assessment examines the implications of these principles.

Return to Top

The Future

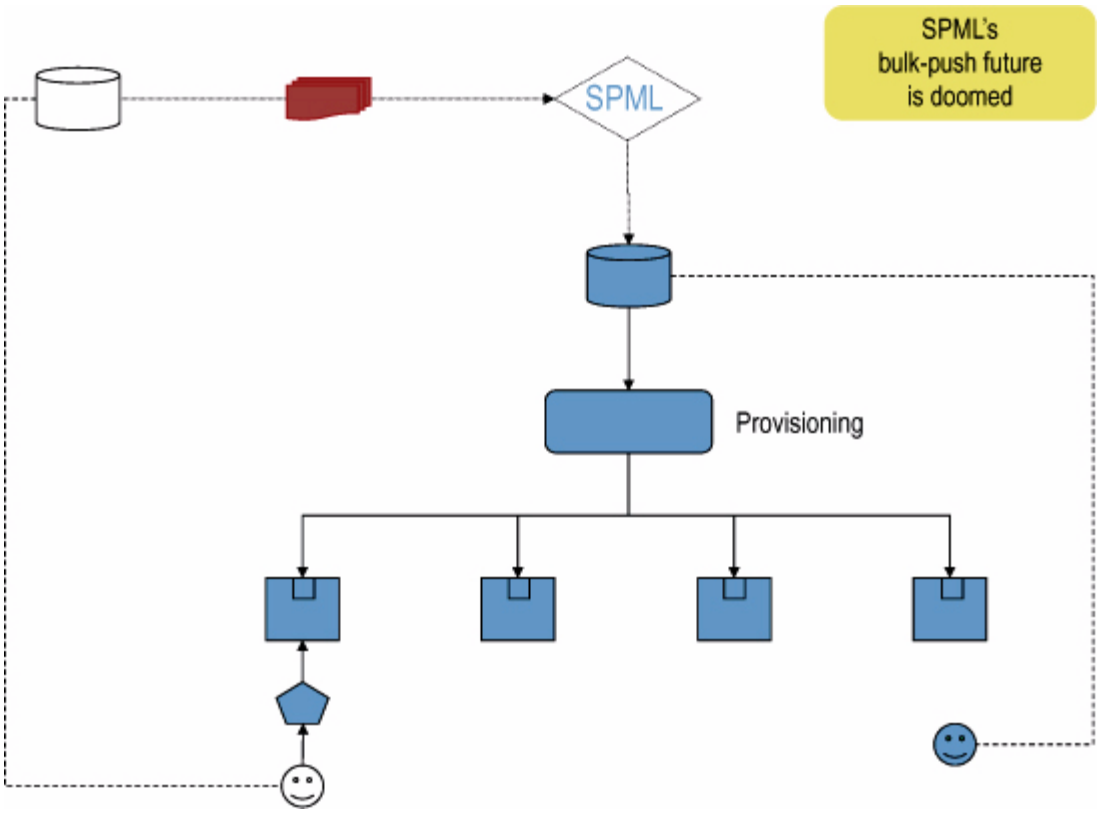
The future of IdM will be driven toward a pull model by the compelling business advantages of an open identity market, which can provide high-quality identities at low cost. Burton Group believes the IdM landscape will evolve toward a market for identities supported by a pull-model infrastructure in two phases.

The first phase is under way, but it is developing alongside two deficient options. Figure 10 illustrates the use of SPML to extend provisioning to a federated environment. This architecture perpetuates the market for IdM systems and does nothing to create the market for identities that the modern enterprise requires.

The model is also complex to administer and inefficient to operate: It moves a lot of data to endpoints where the data will never be used or where the data will become obsolete and have to be updated by another push before being used. It doesn't deal gracefully with situations in which multiple third-party providers possess information about overlapping user populations. It's hard to optimize — the identity data sources don't know in advance what attributes will be required by a particular endpoint for a particular transaction, so they aren't able to omit unnecessary attributes.

Given these deficiencies, it's not surprising that, as Mark Diodati has observed in the upcoming Burton Group technical case study "OASIS or Mirage: Standards-Based Provisioning," SPML adoption is lagging and the technical community's support for evolving the standard has not been enthusiastic.

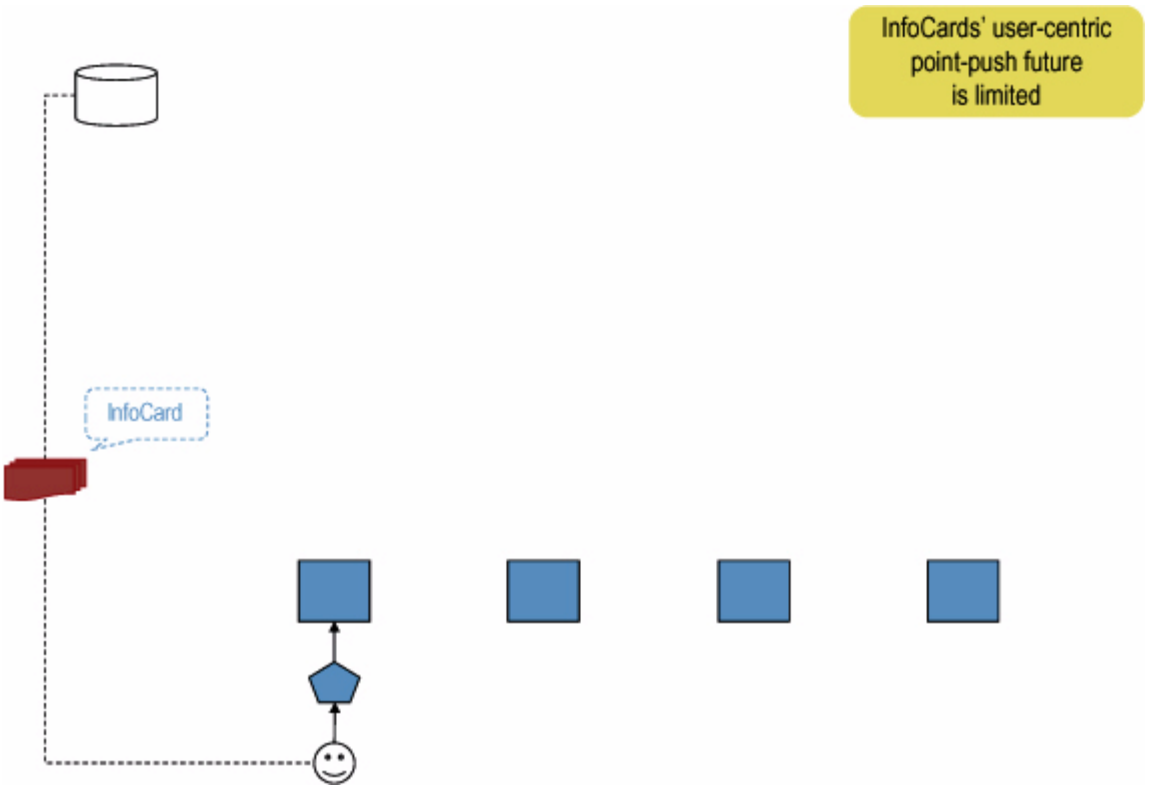
Figure 10. The SPML Evolution of Bulk-Push Won't Solve the Problem



User-centric identity, based on open standards including SAML, WS-Trust, Web Services Federation Language (WS-Federation), and information cards, has been proposed as an alternative to management of identity by enterprises.

But user-centric identity, from the viewpoint of the enterprise, replaces the current problems with new ones rather than solving the current problems. Today's user-centric identity systems are no better than federated provisioning systems at aggregating information from multiple sources to build an identity for a single user; they create a market for high-quality identity providers but do nothing to address the fact that no provider can supply high-quality identities for all individuals whom the modern enterprise needs to identify.

Figure 11. User-Centric Identity Will Persist But Not Dominate

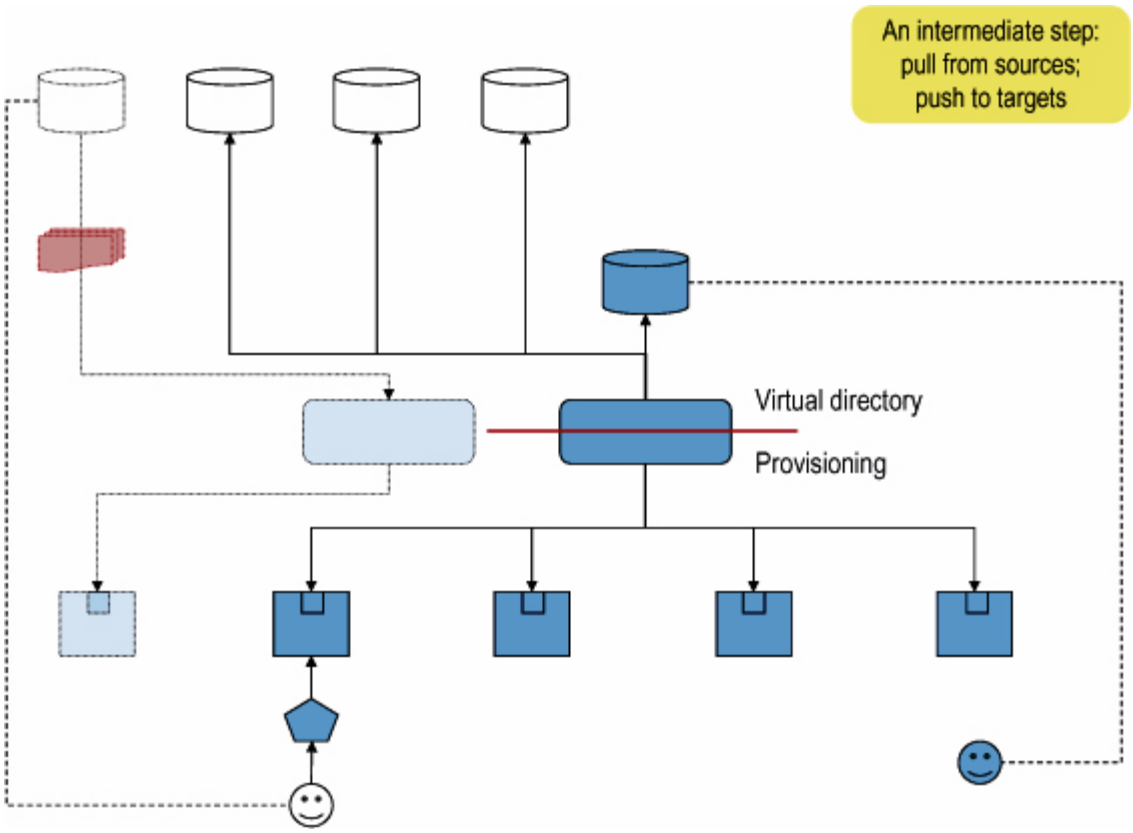


The technologies underlying user-centric identity will not go away. Indeed, they will be critical elements of the foundation of pull-model identity systems. And these technologies will continue to be used in user-centric configurations for the foreseeable future, particularly in business scenarios that can be solved by a single identity provider serving stable sets of identity attributes. But Burton Group expects the mainstream enterprise IdM architecture to evolve away from a user-centric, front-door push model and toward a back-end pull model.

The first step in this evolution will look like Figure 12. In this hybrid push-pull model, the provisioning engine and its connectors to applications remain unchanged. Integration of off-premises third-party identity sources is accomplished through the introduction of a virtual directory that masks the difference between these sources and the on-premises employee database.

Not all third-party sources will be set up to accept pull-protocol requests right away, so the hybrid architecture will probably exist alongside SPML-based, push-protocol federated provisioning solutions for several years.

Figure 12. A Hybrid Push-Pull Model Is a Steppingstone



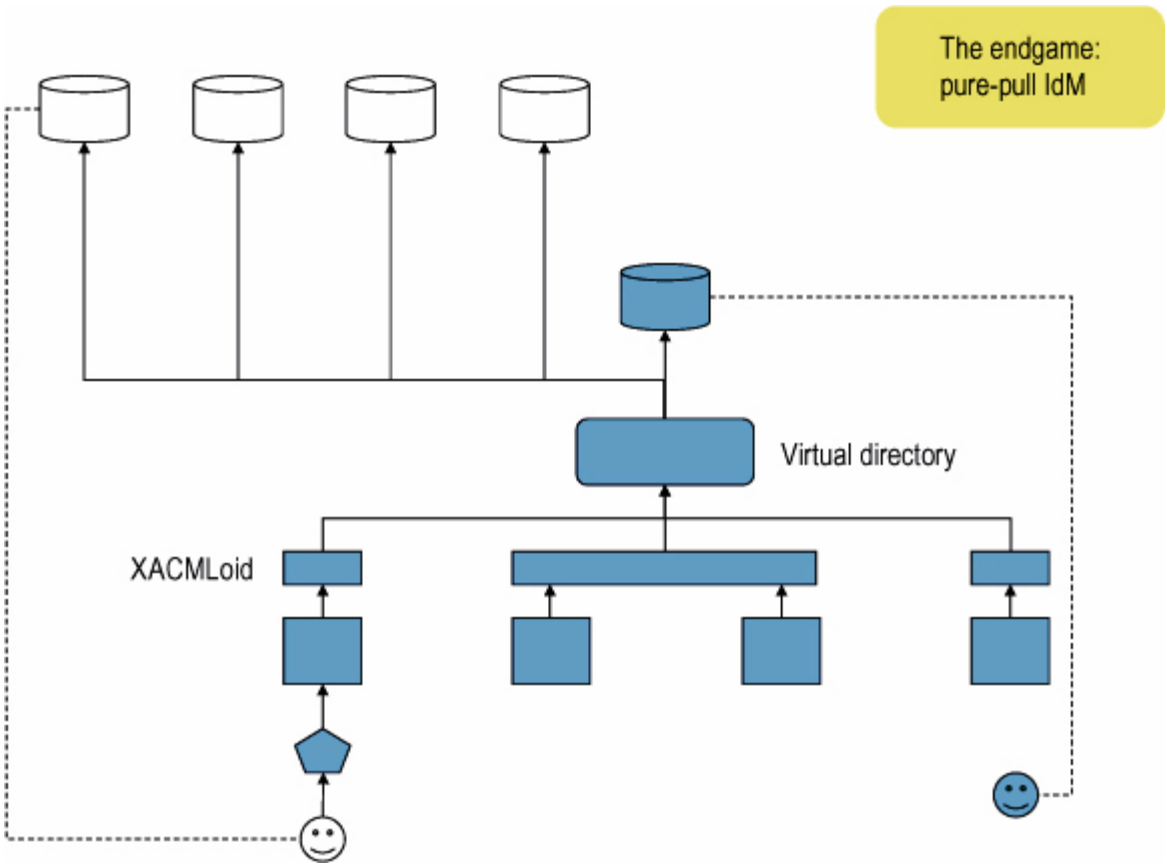
This hybrid solution still suffers from the excessive data movement problems caused by the push model. The virtual directory will have to retrieve information about multiple (often many) individuals at a time in order to satisfy the requests of the provisioning engine. There are other problems, too: Virtual directory protocols won't support the notifications necessary to alert the provisioning engine to changes in the profile of a user whose identity is stored remotely; some additional protocol support will have to be provided to trigger provisioning. When a new user shows up at the enterprise's front door via a federated logon request, the provisioning engine won't automatically know which repository should be used as the user's source of identity information, so a service similar to Shibboleth's Where Are You From (WAYF) service will need to be provided. And of course, provisioning connectors (with all their problems) will still exist in the hybrid environment.

But the hybrid environment has two critical advantages. First, the virtual directory provides an interface abstraction that hides the difference between local and remote sources of identity information. The virtual directory interface is the implementation of Siegel's principle that "names matter, locations don't." By translating names (of users) into locations (of providers serving their identity information), the virtual directory interface ensures that the enterprise doesn't get locked into a single provider. It serves as the boundary separating production of identities from consumption of identities, or to put it another way, it serves as the boundary between the enterprise and all identity providers-including on-premises identity providers.

Second, when the virtual directory and protocols that permit pull-model access to identity information are in place, Siegel's second criterion will have been met: Identity data will have migrated from the "deep web" (i.e., proprietary silos in individual enterprises and identity providers' walled gardens) to the "open web" (i.e., provider systems accessible via open protocols and interfaces). Enablement of identity providers for access via virtual directory interfaces thus enables identity producers to support pull-model access and thus to compete in a market for identities.

Once a market of pull-enabled providers exists, enterprises will have a strong incentive to take advantage of price and quality advantages created by competition among the providers. But until enterprises implement the architecture illustrated in Figure 13, they will be stuck in the identity providers market instead of the identities market.

Figure 13. A Pure-Pull IdM Architecture



As long as an enterprise still needs to push identity data from a provisioning engine to endpoint systems via connectors, it will have to choose its identity providers well before the time a user requests a transaction. This means the enterprise has to buy identities in advance, without knowledge of the cost or quality that will be available at the time (i.e., sometime in the future) the information is used in a transaction.

To get the best combination of identity cost and quality for each transaction, the enterprise needs to be able to wait until the moment before the transaction to decide which provider it will use as the source of identity data to support the transaction. In other words, the enterprise needs to ensure that Siegel's third principle — data does not move — is satisfied until the moment the data is required.

Doing this means building applications that assume identity information will be retrieved at the time of the transaction, not preconfigured through local administrative interfaces.

Application architectures that support this are already available; Microsoft's Geneva is one example. Application frameworks are increasingly being designed, like Geneva, with the ability to externalize authorization. Externalizing authorization means calling out to an external policy decision point (PDP) at the time a transaction is requested to see whether the transaction should be allowed.

But externalized authorization isn't the only technology needed to eliminate provisioning connectors. The other piece of the puzzle is a PDP that supports "contextual authorization" — the ability to call out to an external source to retrieve attributes it needs to make a decision. PDPs supporting contextual authorization exist today, too: They are often called "entitlements management systems." This name is unfortunate because these systems don't manage entitlements. Instead, they make fine-grained access decisions. In the diagram, they're referred to as "XACMLoids" (many, but not all, of these systems are based on the Extensible Access Control Markup Language [XACML] standard) in order to avoid the confusion that might be caused by calling them "entitlements management."

Figure 13 thus represents a "pure-pull" IdM system. The user authenticates himself via a federation protocol and provides a user name and some WAYF information describing what identity providers know about him. The application calls out via externalized authentication interfaces to a XACMLoid, which in turn calls the virtual directory interface to retrieve the attributes it needs to make a decision. The virtual directory executes some logic to determine which provider or set of providers has the best mix of low-cost and high-identity assurance for the particular attributes of the particular user in question, and it retrieves those attributes from the chosen provider(s). Note that the virtual directory's implementation need not use directory protocols to retrieve attributes from providers. It may use SAML or information-card protocols to request attributes (some information card implementers already support this kind of back-end access to attributes). The virtual directory then passes

the attributes back to the XACMLoid, which makes a decision and passes the decision to the application, which enforces the decision and, if appropriate, executes the transaction.

This model has a few important advantages.

The first is that identity lifecycles of non-employees don't have to be visible to the enterprise. As long as an identity provider knows when an individual's status changes, the provider can update the attributes, and the enterprise will always get current attributes because it always retrieves current information at the time of use (instead of storing it locally and running the risk that it has changed without appropriate notification).

A second advantage is that this model provides much better support for "identity oracles" (described in the Burton Group overview "A Relationship Layer for the Web . . . and for Enterprises, Too") than a push model. An identity oracle manages identity information but does not distribute it. Instead, it answers questions based on the identity information. For example, an application might ask an identity oracle "Is Bob's credit rating high enough to justify approving a $5,000 loan in light of underwriting rule X?" and the oracle might check Bob's credit rating in the light of underwriting rule X and answer "yes." The pure-pull model supports this interaction because it waits until it knows the details of the transaction before using any identity information. The push model, which needs to ship information in advance, can't know whether Bob will ask for $5,000 or $50,000 — and so it has to send Bob's credit rating to the application. Identity oracles can improve privacy protection for users, and they can also be used to build multi-party business models that can't be supported if all parties have to share information.

The benefits of a pure-pull model won't come without cost or effort. A pure-pull model with third-party providers requires stronger and more explicit governance processes than one with a monolithic in-house provider. Policies governing use of outside identity providers for access to internal systems are not yet well articulated; efforts such as the Trust Framework Provider Initiative will go some way to filling in this gap. A lot of application rework will be necessary to support externalized authorization and contextual authorization. Connecting an on-premises virtual directory interface to a variety of back-end providers and providing logic that dynamically selects providers will not be simple.

Finally, the enterprise IT department will have to adapt from being both the consumer and provider of all identities to being the consumer of all and the provider of only some identities.

Those changes won't happen overnight. Burton Group expects the transition from push-model IdM to a pull-model identity market to take five to seven years.

Return to Top

Strengths

Strengths of a pull-model IdM environment include:

- **Cost:** Identities and attributes can be sourced individually from the lowest-cost provider.
- **Quality:** Identities and attributes can be sourced individually from the highest-quality provider.
- **Aggregation:** Attributes from multiple sources can easily be combined into a single identity.
- **Authority:** Each attribute can be drawn directly from its authoritative source.
- **Timeliness:** Identities and attributes are always retrieved from an authoritative repository at the time of use, eliminating problems of stale information and failure to recognize and react to identity lifecycle events.
- **Integration:** Providers in a pull-model environment are accessed using standard interfaces and protocols, thereby facilitating quick, simple, and cheap integration of new sources.
- **Provider independence:** Because data does not move, switching providers does not involve migration. Because providers are accessed using open-standard protocols, switching providers does not involve integration work. Vendor advantage is based solely on cost and quality.

Return to Top

Weaknesses

Weaknesses of a pure-pull IdM environment include:

- **Trust:** Sourcing identity information from third-party providers raises significant trust issues; identity

assurance practices, audits, regulations, and business terms for third-party identity providers are not yet mature.

- **Accountability:** When a transaction fails because of inaccurate identity information provided by a third party, no clear legal or contractual remedies are available to the enterprise that relied on the information.
- **Standards:** Some standards necessary to support a pure-pull IdM model are not yet complete; in particular, identity assurance standards are still being formulated.
- **Infrastructure support:** Significant infrastructure changes are required to support a pull-model identity environment; these changes will take years to implement and deploy.

Return to Top

Recommendations

Although the pure-pull model of identity management (IdM) probably won't become dominant for five to seven years, enterprises that don't prepare for a timely move to a pull model may put themselves at significant competitive disadvantage.

Burton Group recommends tackling the transition using the following roadmap:

1. **Divide production of identities from consumption:** Introduce a virtual directory interface between your provisioning engine and your on-premises authoritative identity source or sources.
2. **Build an identity consumption policy:** Make your identity assurance criteria explicit. Build a catalog of "who's authoritative for what attributes" and establish a governance process for keeping this catalog up to date. Determine what business terms and conditions will be required of external identity providers — pay special attention to quality of service issues, remedies in case of failure, and compliance and access certification requirements. If other organizations are depending on you for identity information, be sure to examine their requirements in detail.
3. **Build support for multiple providers:** Consider third-party providers for some identities (e.g., contractors, partners, and customers) or for some attributes (e.g., professional certification status, citizenship, and residency status). When selecting providers, pay careful attention to business terms, identity assurance, cost of identity, and the willingness and ability of the provider to serve identity information in real time over standard protocols. If necessary, integrate protocols necessary to support third-party identity provision into the back end of your virtual directory.
4. **Externalize authorization from applications:** Build or acquire new applications that call out for authorization decisions rather than embedding authorization in application code. Speak to your application-platform and development-tool providers early in the process to gauge their willingness and ability to support this capability.
5. **Contextualize authorization:** Use Extensible Access Control Markup Language (XACML) or similar rules engines to build authorization policies that can retrieve identity attributes at runtime based on the user's identifier. Don't build policies which assume that identity attributes will be passed into the policy decision point (PDP) by the application.
6. **Phase out connectors:** Over time, phase out applications that don't support pull-protocol identity and externalized authorization. Decommission connectors as appropriate. Eventually, aim to phase out push-model provisioning.

Return to Top

Burton Group was recently acquired by Gartner, Inc., the world's leading information technology research and advisory company.